Kotlin - For Loop

What are loops?

Imagine a situation when you need to print a sentence 20 times on your screen. You can do it by using print statement 20 times. What about if you need to print the same sentence one thousand times? This is where we need to use loops to simplify the programming job. Actually, Loops are used in programming to repeat a specific block of code until certain condition is met.

Kotlin supports various types of loops and in this chapter we are going to learn Kotlin **for** loop.

Kotlin For Loop

Kotlin **for** loop iterates through anything that provides an iterator ie. that contains a countable number of values, for example arrays, ranges, maps or any other collection available in Kotlin. Kotlin **for** loop is equivalent to the **foreach** loop in languages like C#.

> Kotlin does not provide a conventional **for** loop which is available in C, C++ and Java etc.

Syntax

The syntax of the Kotlin **for** loop is as follows:

```
for (item in collection) {
    // body of loop
}
```

Iterate Through a Range

We will study Kotlin Ranges in a separate chapter, for now you should know that Kotlin Ranges provide iterator, so we can iterate through a range using **for** loop.

Following is an example where the loop iterates through the range and prints individual item. To iterate over a range of numbers, we will use a range expression:

```
fun main(args: Array<String>) {
   for (item in 1..5) {
      println(item)
   }
}
```

When you run the above Kotlin program, it will generate the following output:

```
1
2
3
4
5
```

Let's see one more example where the loop will iterate through another range, but this time it will step down instead of stepping up as in the above example:

```
fun main(args: Array<String>) {
   for (item in 5 downTo 1 step 2) {
      println(item)
   }
}
```

When you run the above Kotlin program, it will generate the following output:

```
5
3
1
```

Iterate Through a Array

Kotlin Array is another data type which provides iterator, so we can use **for** loop to iterate through a Kotlin array in the similar way as we did it for the ranges.

Following is an example where we used **for** loop to iterate through an array of strings:

```
fun main(args: Array<String>) {
   var fruits = arrayOf("Orange", "Apple", "Mango", "Banana")

   for (item in fruits) {
      println(item)
   }
}
```

When you run the above Kotlin program, it will generate the following output:

```
Orange
Apple
Mango
Banana
```

Let's see the same example once again, but this time we will iterate through the array using its index.

```
fun main(args: Array<String>) {
   var fruits = arrayOf("Orange", "Apple", "Mango", "Banana")
```

```
    for (index in fruits.indices) {
        println(fruits[index])
    }
}
```

When you run the above Kotlin program, it will generate the following output:

```
Orange
Apple
Mango
Banana
```

Quiz Time (Interview & Exams Preparation)

Q 1 - Which of the following is true about Kotlin for loop?

A - It is used to loop through an iterator.

B - Kotlin does not provide conventional for loop like C, C++ or Java.

C - Kotlin for loop is equivalent to the foreach loop in languages like C#.

D - All of the above

Q 2 - What will be the last number printed by the following for loop?

```
fun main(args: Array<String>) {
    for (item in 6 downTo 1 step 2) {
        println(item)
    }
}
```

A - 6

B - 5

C - 3

D - 2